**Эйнор Леонард 2-ИП**

**Отчёт по практическому заданию 1**

Необходимо реализовать программу, сжимающую исходный текст файла Input.txt, в результирующий файл output.txt, с предусмотреной возможностью деархивации.

**В проекте задачи содержится 1 файл:**

**Zadacha1.cpp**

Файл содержит основной код со всеми функциями, необходимыми для реализации данного задания.

**Описание функций:**

1. **Void CreateInput()**

Функция создаёт файлы input.txt и output.txt и записывает строку в input.txt.

**Переменные:**

file

enterInt

forInput

enterStr

**Алгоритм**:

Открываем поток для файла input.txt и создаём все необходимые переменные.

Просим пользователя ввести строку для записи в файл, пока в ней есть запрещённые символы.

Если в строке обнаружены запрещённые символы, то вывести ошибку и начать цикл заного, иначе записать строку в переменную для её хранения.

Если файл input.txt открылся, то

Если пользователь хочет его перезаписать, то открыть файл в режиме записи (для очистки) и записать в него строку для записи,

Если пользователь не хочет перезаписывать файл, то файл закрывается без изменений.

Иначе если файл input.txt не открылся, то создать и записать в него строку.

Вывести пользователю содержимое файла input.txt.

Открываем в том же потоке файл output.txt, если его не существует, он создастся.

1. **Void Archive()**

В данной функции производится сжатие input.txt в

output.txt.

**Переменные:**

inputString

outputString

counterSameSimbols

file

i

i1

**Алгоритм**:

Пока количество обработанных символов меньше их количества в вводной строке

Записывать символ в итоговую строку и считать символы за ним, пока не попадётся отличный от начального символ,

Записать посчитанное количество одинаковых символов в итог и начать цикл со следующего отличного символа.

После открыть output.txt для вывода (это его очистит) и записать в него итоговую строку.

В конце вывести вводную и итоговую строки.

В функции реализованны проверки на открытие нужного файла и

вывод ошибок связанных с этим и некоректным вводом.

1. **Void Dearchive()**

В данной функции файл разархивируется в изначальную форму.

**Переменные:**

inputString

outputString

fullNumberStr

fullNumber

file

i

i1

**Алгоритм**:

Пока количество обработанных символов меньше их кол-ва в исходной строке

Если первый символ цифра вывести ошибку и завершить работу функции

(Часть кода для обработки многоразрядных чисел)

Если текущий символ цифра добавить его в строку полного числа и пока следующие символы цифры, продолжать добавлять их в строку полного числа.

Перевести получившуюся строку с полным число в число.

Пока количество поставленных букв меньше нужного количества, добавлять по букве в итог.

Если текущий символ не цифра, добавить символ в итог и присвоить строке полного числа 1 (для корректной работы цикла).

После записать итоговую строку в файл output.txt и вывести строку до преобразования и после.

В функции реализованны проверки на открытие нужного файла и вывод ошибок связанных с этим и некоректным вводом.

1. **int main()**

Эта функция содержит основной код программы реализующий

выбор пользователем нужных ему функций программы.

**Переменные:**

enterInt

enterStr

endProgram

**Алгоритм**:

Спрашивать пользователя о действии, которое необходимо выполниить пока выбор некоректный. В случае некоректного ввода выводить ошибку.

Если ввод корректный, то запускается функция соответствующая выбору пользователя.(1. Создать файлы и заполнить 2. Заархивировать 3. Разархивировать 0. Закрыть программу)

**Код программы**:

#include <Windows.h>

#include <fstream>

#include <locale.h>

#include <iostream>

#include <string>

using namespace std;

void CreateInput() { //Функция создания и ввода в input/output

fstream file("input.txt");

int enterInt = 9;

string forInput, enterStr = "0";

while (enterStr.find\_first\_of("0123456789") < -1) { //Пока в строке есть цифры

cout << "Введите строку для записи в файл = ";

getline(cin, enterStr);

cout << endl << endl;

if (enterStr.find\_first\_of("0123456789") < -1) { //Вывод ошибки при обнаружении чисел

cout << "\x1b[31mОшибка 0: Некоректный ввод данных\x1b[0m" << endl;

} else { //Сохранение строки в другой переменной и очищение вводной строки для дальнейшего использования

forInput = enterStr;

enterStr.clear();

} }

if (file) { //Если input открылся

cout << "Файл input уже существует, хотите перезаписать его?" << endl;

while (enterInt != 1 and enterInt != 2) { //Пока выбор пользователя некоректный

cout << "1 - Да" << endl << "2 - Нет" << endl << "Ваш выбор = ";

getline(cin, enterStr);

cout << endl << endl;

if (enterStr.find\_last\_not\_of("12") != -1 or enterStr.length() != 1) { //Проверка на буквы

cout << "\x1b[31mОшибка 0: Некоректный ввод данных\x1b[0m\n";

} else {

enterInt = atof(enterStr.c\_str());

}

switch (enterInt) { //Переключатель выбора пользователя

case 1: {

file.close();

file.open("input.txt", ios::out);

file << forInput; file.close();

cout << "\n" << "\x1b[32mФайл успешно перезаписан\x1b[0m" << "\n\n\n\n";

break;

} //Перезаписать input

case 2: {

file.close();

cout << "\n" << "\x1b[33mФайл остался без изменений\x1b[0m" << "\n\n\n\n";

break;

}}}} else { //Если input не открылся - создать

file.open("input.txt", ios::in | ios::app);

file << forInput;

file.close();

cout << "\n" << "\x1b[32mФайл успешно записан\x1b[0m" << "\n\n\n\n";

}

file.open("input.txt", ios::in);

getline(file, forInput);

cout << "\n\x1b[33mВ файле input.txt находится: \x1b[0m\n" << forInput << "\n\n\n\n";

file.close();

file.open("output.txt", ios::in | ios::app);

file.close();

cout << "\n" << "\x1b[32moutput.txt успешно создан\x1b[0m" << "\n\n\n\n";

}

void Archive() { //Функция архивирования строки

string inputString, outputString;

fstream file("input.txt"); //Открыть файла ввода

if (!file) { //Проверка открытия файла

cout << "\n" << "\x1b[31mОшибка 1: Необходимые файлы не открылись. Попробуйте запустить команду 1\x1b[0m" << "\n\n\n\n";

return;

}

getline(file, inputString); //Считать строку из файла

file.close(); //Закрыть файл

for (int i = 0, counterSameSimbols; i < inputString.length(); i += counterSameSimbols) { //Цикл получающий каждый раз новый символ

counterSameSimbols = 1;

outputString += inputString[i]; //Запись символа в итоговую строку

for (int i1 = 1; inputString[i] == inputString[i + i1]; i1++, counterSameSimbols++) {} //Счётчик количества одинакового символа

if (counterSameSimbols > 1) { //Если повторов символа больше 1, то записывать цифру

outputString += to\_string(counterSameSimbols);

}}

file.open("output.txt", ios::out); //Открытие файла вывода

if (!file) { //Проверка открытия файла

cout << "\n" << "\x1b[31mОшибка 1: Необходимые файлы не открылись. Попробуйте запустить команду 1\x1b[0m" << "\n\n\n\n";

return;

}

file << outputString; //Записать итог в файл

file.close(); //Закрыть файл

cout << "\n" << "\x1b[32mАрхивация успешно выполнена\x1b[0m" << "\n\n\n\n";

cout << "\n<<<<<<<<<<<<<<<<<<<<<<<<<<\n" << inputString;

cout << "\n==========================";

cout << "\n>>>>>>>>>>>>>>>>>>>>>>>>>>\n" << outputString << "\n\n\n";

}

void Dearchive() {

string inputString, fullNumberStr, outputString;

int fullNumber;

fstream file("output.txt", ios::in);

if (!file) { //Проверка открытия файла

cout << "\n" << "\x1b[31mОшибка 1: Необходимые файлы не открылись. Попробуйте запустить команду 1\x1b[0m" << "\n\n\n\n";

return;

}

getline(file, inputString);

file.close();

for (int i = 0; i < inputString.length(); i += fullNumberStr.length()) { //Цикл перебирает буквы и числа

fullNumberStr.clear(); //Очистка fullNumberStr после предыдущей итерации

if (inputString[i] >= '0' and inputString[i] <= '9' and i == 0) { //Если первый символ цифра

cout << "\n" << "\x1b[31mОшибка 2: Первый символ цифра\x1b[0m" << "\n\n\n\n";

return;

}

if (inputString[i] >= '0' and inputString[i] <= '9') { //Если символ - цифра

fullNumberStr += inputString[i]; //добавляем к строке fullNumberStr текущее число

for (int i1 = 1; inputString[i + i1] >= '0' and inputString[i + i1] <= '9'; i1++) { //Пока текущий символ - число

fullNumberStr += inputString[i + i1]; //Добавлять число в строку полного числа

}

fullNumber = stoi(fullNumberStr); //перевод всех чисел в единую цифру

for (int i1 = 1; i1 < fullNumber; i1++) { //Пока количество букв меньше уже поставленных

outputString += inputString[i - 1]; //Добавлять букву в итог

}} else {

outputString += inputString[i]; //Добавить текущую букву в итог

fullNumberStr = '1';

}}

file.open("output.txt", ios::out);

if (!file) { //Проверка открытия файла

cout << "\n" << "\x1b[31mОшибка 1: Необходимые файлы не открылись. Попробуйте запустить команду 1\x1b[0m" << "\n\n\n\n";

return;

}

file << outputString; //Записать итог в файл

file.close();

cout << "\n" << "\x1b[32mДеархивация успешно выполнена\x1b[0m" << "\n\n\n\n";

cout << "\n<<<<<<<<<<<<<<<<<<<<<<<<<<\n" << inputString;

cout << "\n==========================";

cout << "\n>>>>>>>>>>>>>>>>>>>>>>>>>>\n" << outputString << "\n\n\n";

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int enterInt, endProgram = 0;

string enterStr;

while (endProgram == 0) { //Цикл работы программы

cout << "1 - Создать файлы и записать строку" << endl << "2 - Заархивировать файл" << endl << "3 - Разархивировать файл" << endl << "0 - Выход из программы" << endl << "Ваш выбор = ";

getline(cin, enterStr);

cout << endl << endl;

if ((enterStr.find\_last\_not\_of("0123") != -1) or (enterStr.empty())) { cout << "\x1b[31mОшибка 0: Некоректный ввод данных\x1b[0m" << endl; continue; }

else { enterInt = atof(enterStr.c\_str()); }

if (enterInt < 0 or enterInt > 3) { cout << "\x1b[31mОшибка 0: Некоректный ввод данных\x1b[0m" << endl; continue; }

switch (enterInt) { //Переключатель выбора пользователя

case 1: { CreateInput(); break; } //Запуск функции создания файлов

case 2: { Archive(); break; } //Запуск функции зашифровки файлов

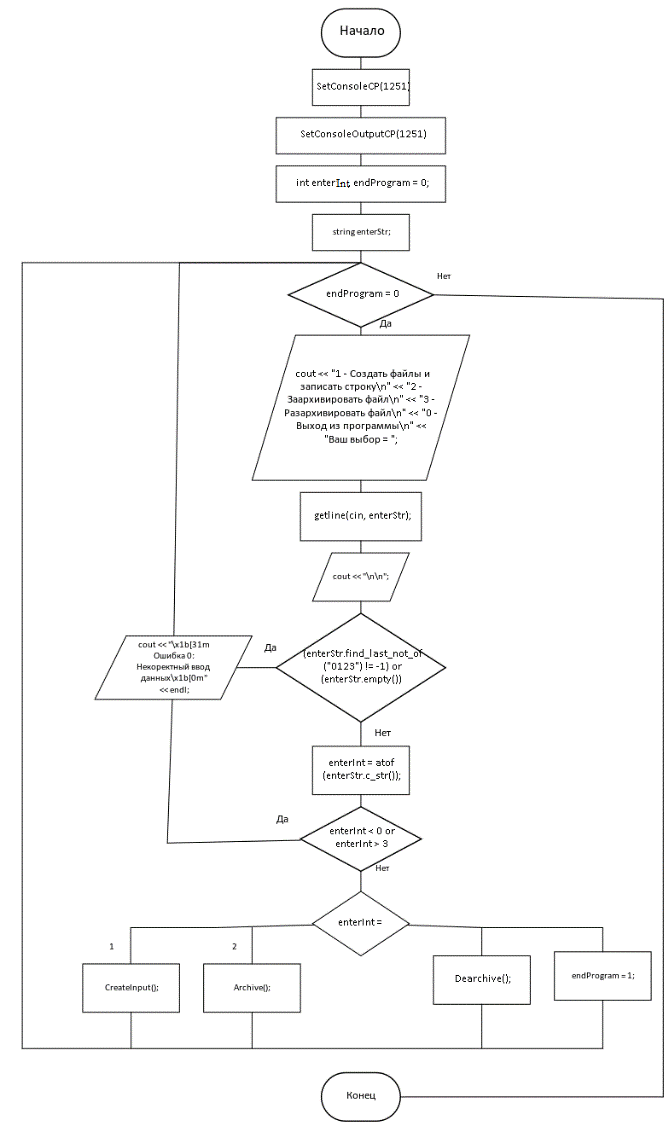
case 3: { Dearchive(); break; } //Запуск функции расшифровки файла

case 0: { endProgram = 1; break; } //Выключение программы

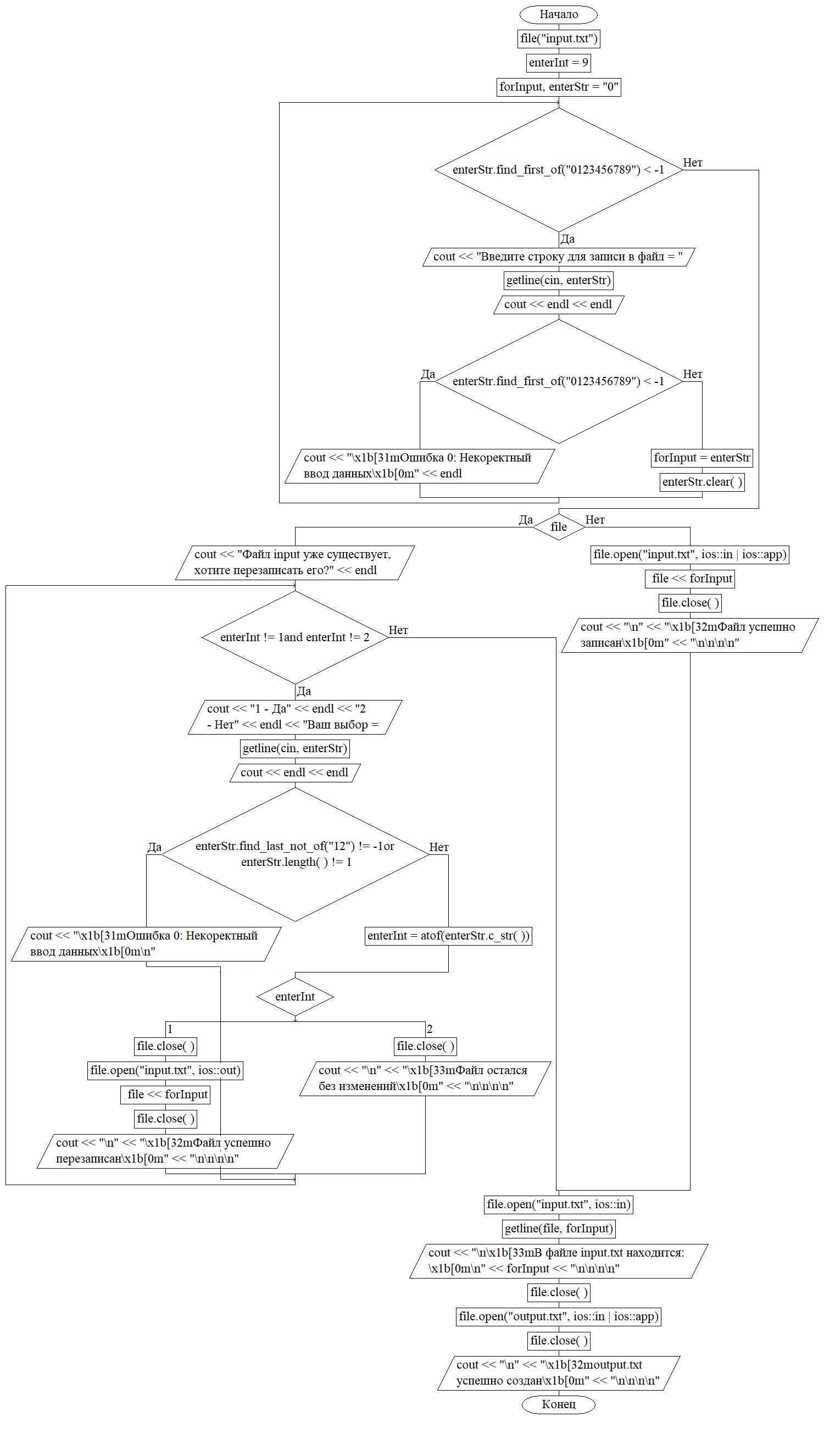
}}}

**Блок-схемы функций**

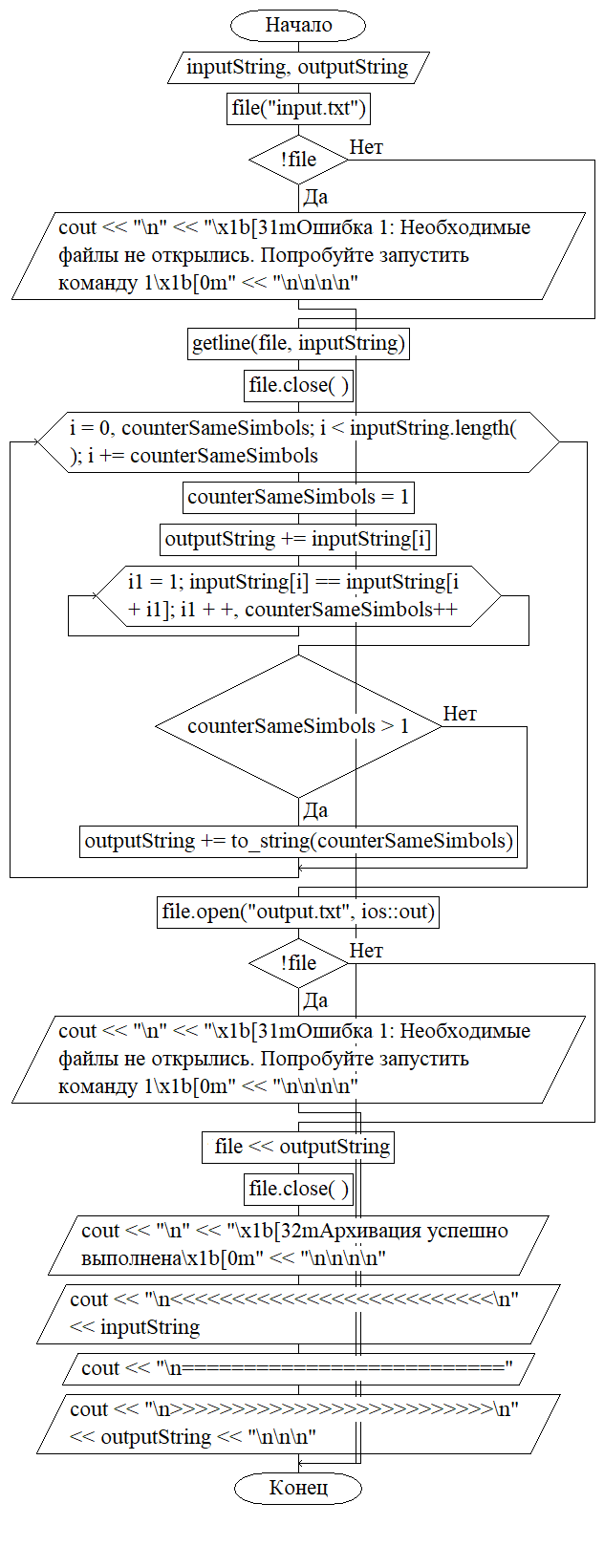
**Int Main()**



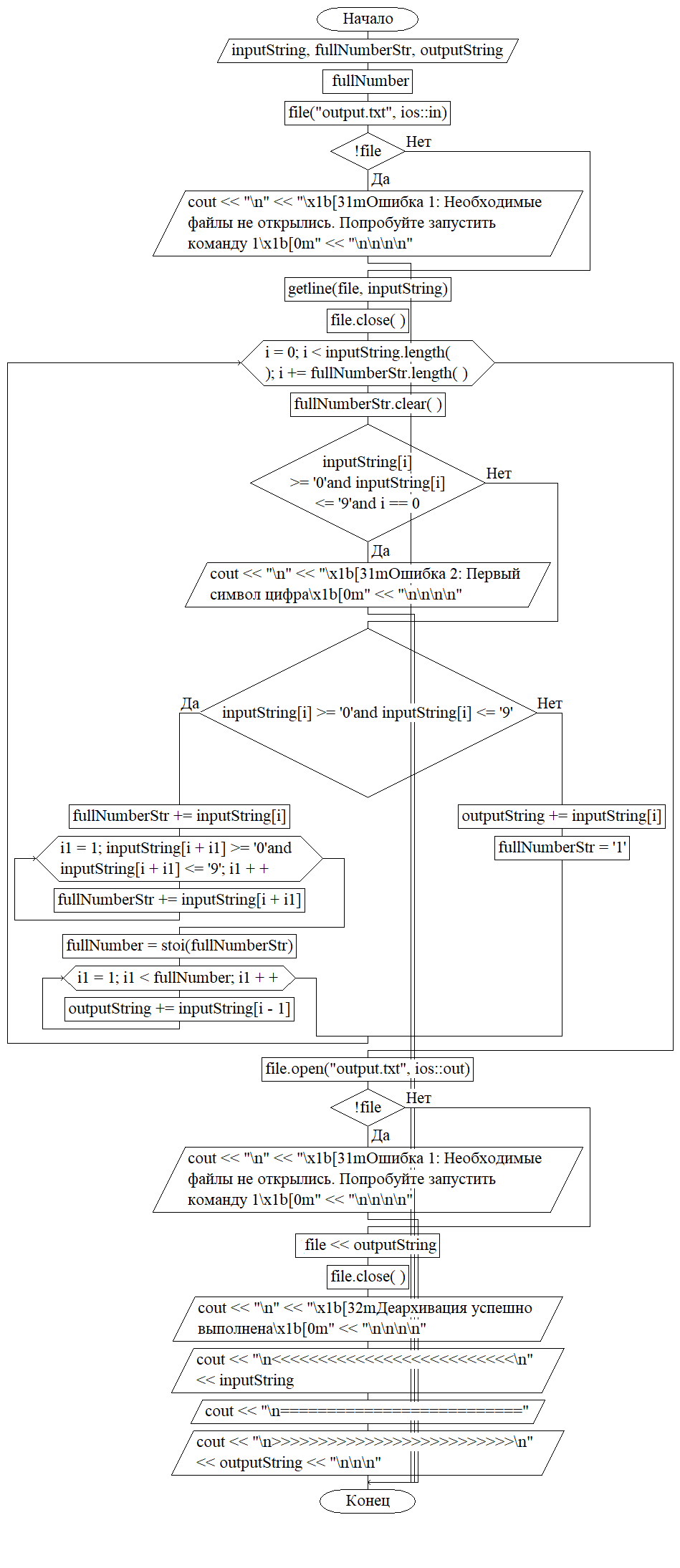
**Void CreateInput()**

****

**Void Archive()**

****

**Void Dearchive()**

****

**Отчёт по практическому заданию 2**

Необходимо реализовать программу, сжимающую исходный текст файла Input.txt, в результирующий файл output.txt, с предусмотреной возможностью деархивации.

**В проекте задачи содержится 1 файл:**

**Zadacha2.cpp**

Файл содержит основной код со всеми функциями, необходимыми для реализации данного задания.

**Описание функций:**

1. **Void InputS(string s, Post& A)**

Функция проверяет переданную в неё строку и если строка корректна, то

записывает эту строку в переданный экземпляр структуры.

**Переменные:**

s

A

**Алгоритм**:

Если строка пустая, не содержит хотя бы одной цифры и одного знака или содержит запрещённые символы, то вывести ошибку и закончить работу функции.

Иначе записать строку в A.input и вывести сообщение об успешной записи.

1. **Void OutputS(Post& A)**

Функция преобразовывает пример из A.input в постфиксную форму,

записывая её в A.output.

**Переменные:**

A

output

brackets

i

operands

**Алгоритм:**

Если переменная input экземпляра A пуста, то вывести ошибку и завершить работу функции.

Цикл перебирает пример по символу слева на право, пока пример не кончился.

Если входящий символ число, то записать его в итог.

Если входящий символ операция, то если стек пуст или содержит на вершине открывающуюся скобку, то добавить символ в стек, иначе добавить вершину стека в итог и добавить в стек входящую операцию.

Если входящий символ открывающаяся скобка, то добавить в стек входящую операцию и увеличить счётчик скобок на 1.

Если входящий символ закрывающая скобка, то пока не будет встречена открывающая скобка выгружать стек в итог, после удалить из стека открывающуюся скобку.

После обработки всего примера выгрузить оставшиеся в стеке операции в итог.

Записать итоговую строку в структуру.

В функции присутствует защита от закрывающейся скобки без открывающейся.

1. **Void Display(Post& A)**

Функция выводит пример в инфиксной и постфиксной форме

находящийся в переменных input и output переданной структуры.

**Переменные:**

A

**Алгоритм:**

Если переменные структуры A output или input пусты, то вывести ошибку и завершить работу функции,

Иначе вывести пример в инфиксной и постфиксной форме

находящийся в переменных input и output переданной структуры.

1. **Int main()**

Эта функция содержит основной код программы реализующий выбор

пользователем нужных ему функций программы.

**Переменные:**

enterInt

enterStr

endProgram

**Алгоритм**:

Спрашивать пользователя о действии, которое необходимо выполниить пока выбор некоректный. В случае некоректного ввода выводить ошибку.

Если ввод корректный, то запускается функция соответствующая выбору пользователя.(1. Создать файлы и заполнить 2. Заархивировать 3. Разархивировать 0. Закрыть программу)

**Код программы**:

#include <iostream>

#include <string>

#include <Windows.h>

#include <stack>

using namespace std;

struct Post { //Создание структуры и экземпляра

string input;

string output;

}A;

void InputS(string s, Post& A) { //Функция проверяющая переданную строку на корректность и помещающая её в структуру

if (s.empty() or s.find\_first\_of("-+\*:") == -1 or s.find\_first\_of("0123456789") == -1) {

cout << "\x1b[31mОшибка 0: Некоректный ввод данных\x1b[0m\n";

} else {

A.input = s;

cout << "\n\n\n\x1b[32mПример успешно записан!\x1b[0m\n\n\n\n\n";

}}

void OutputS(Post& A) { //Функция перевода input в постфиксную форму

if (A.input.empty()) {

cout << "\x1b[31mОшибка 404: Пример не найден. Попробуйте выбрать ввод примера в структуру\x1b[0m\n";

return;

}

stack <char> operands; //Объявление стека

string output;

int brackets = 0;

for (int i = 0; i < A.input.length(); i++) { //Цикл считывает пример по символу

if (isdigit(A.input[i])) { //Если символ цифра

output += A.input[i];

}

if (A.input[i] == '+' or A.input[i] == '-' or A.input[i] == '\*' or A.input[i] == ':') { //Если символ знак

if (operands.empty() or operands.top() == '(') { //Если стек пуст или на его выршине открывающаяся скобка

operands.push(A.input[i]);

}

else {

output += operands.top();

operands.pop();

operands.push(A.input[i]);

}}

if (A.input[i] == '(') { //Если символ открывающаяся скобка

operands.push(A.input[i]);

brackets++;

}

if (A.input[i] == ')') { //Если символ закрывающаяся скобка

brackets--;

if (brackets < 0) { //Если закрывающаяся скобка идет без открывающей

cout << "\x1b[31mОшибка 1: Встречена закрывающая скобка без открывающей. Проверьте корректность примера\x1b[0m\n";

return;

}

while(operands.top() != '(') { //Пока в стеке не найдена открывающаяся скобка, добавлять элементы стека в итог

output += operands.top();

operands.pop();

}

operands.pop();

}}

while (!operands.empty()) { //Пока стек не пуст, добавлять символы из него в итог

output += operands.top();

operands.pop();

}

A.output.clear(); //Очистка output, на случай если в нём что-то было

A.output = output; //Запись итоговой строки в структуру

cout << "\n\n\n\x1b[32mПреобразование успешно выполнено\x1b[0m\n\n\n\n\n";

}

void Display(Post& A) { //Функция выводит поля структуры

if (A.input.empty() or A.output.empty()) {

cout << "\x1b[31mОшибка 2: Инфиксная или постфиксная форма записи отсутствует, попробуйте последовательно запустить команды 1 и 2\x1b[0m\n";

}

else {

cout << "\n\n\n\x1b[32mВаша структура:\nA.input = " << A.input << "\nA.output = " << A.output << "\x1b[0m\n\n\n\n\n";

}}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int enterInt, endProgram = 0;

string enterStr;

while (endProgram == 0) { //Цикл работы программы

cout << "1 - Ввести пример в структуру\n" << "2 - Перевести пример в постфиксную форму\n" << "3 - Вывести input и output\n" << "0 - Выход из программы\n" << "Ваш выбор = ";

getline(cin, enterStr);

cout << "\n\n";

if ((enterStr.find\_last\_not\_of("0123") != -1) or (enterStr.empty())) { //Если в строке обнаружены некоректные символы

cout << "\x1b[31mОшибка 0: Некоректный ввод данных\x1b[0m\n";

continue;

} else {

enterInt = atof(enterStr.c\_str());

}

if (enterInt < 0 or enterInt > 3) { //Если введенное число недопустимо

cout << "\x1b[31mОшибка 0: Некоректный ввод данных\x1b[0m\n";

continue;

}

switch (enterInt) { //Переключатель выбора пользователя

case 1: { //Запуск функции ввода примера в структуру

cout << "Введите пример = ";

getline(cin, enterStr);

InputS(enterStr, A);

break;

}

case 2: { //Запуск функции перевода в постфиксную форму

OutputS(A);

break;

}

case 3: { //Запуск функции отображения структуры

Display(A);

break;

}

case 0: { //Выключение программы

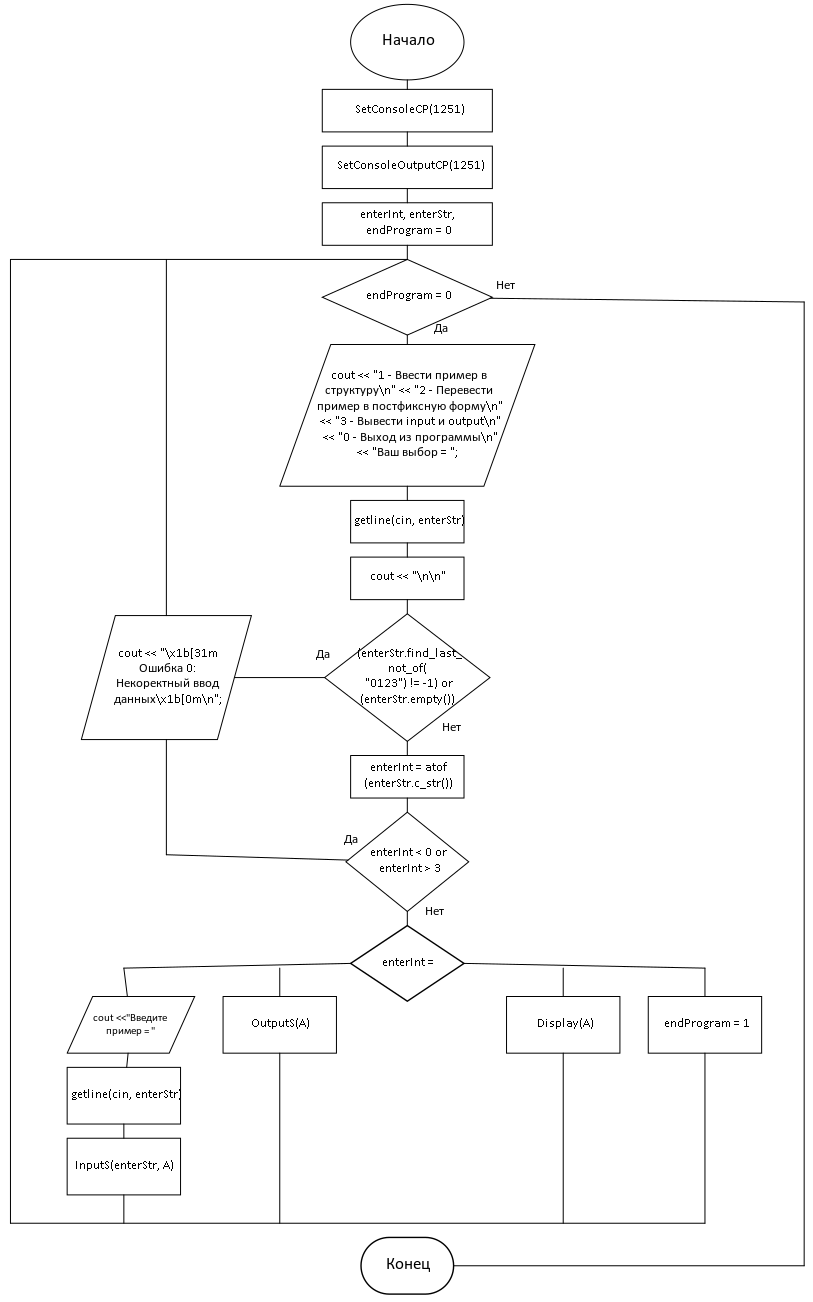
endProgram = 1;

break;

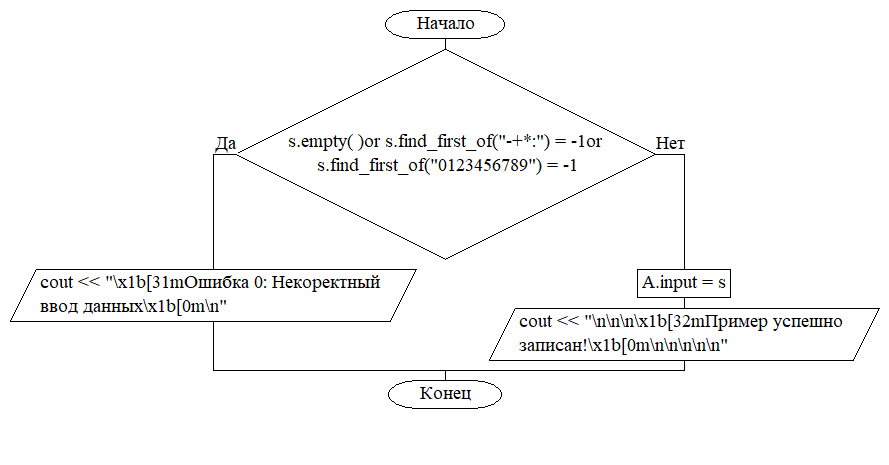
}}}}

**Блок-схемы функций**

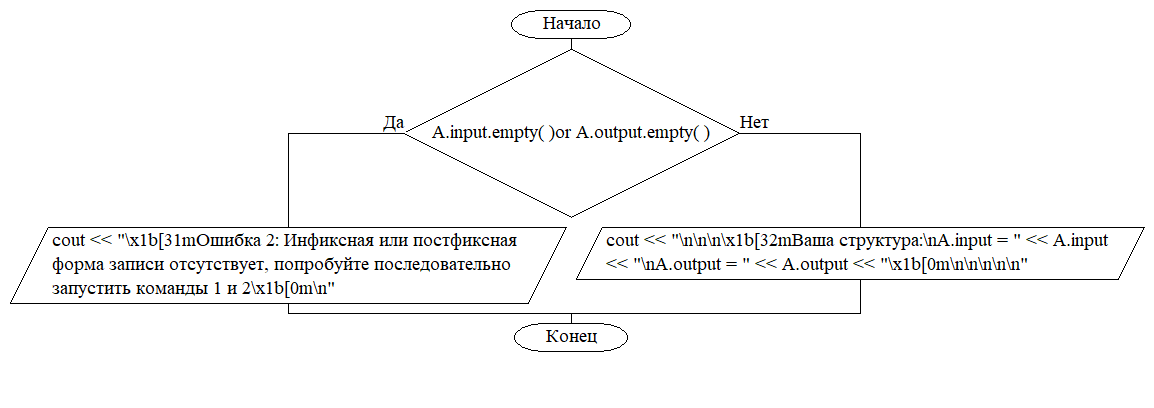
**Int Main()**

****

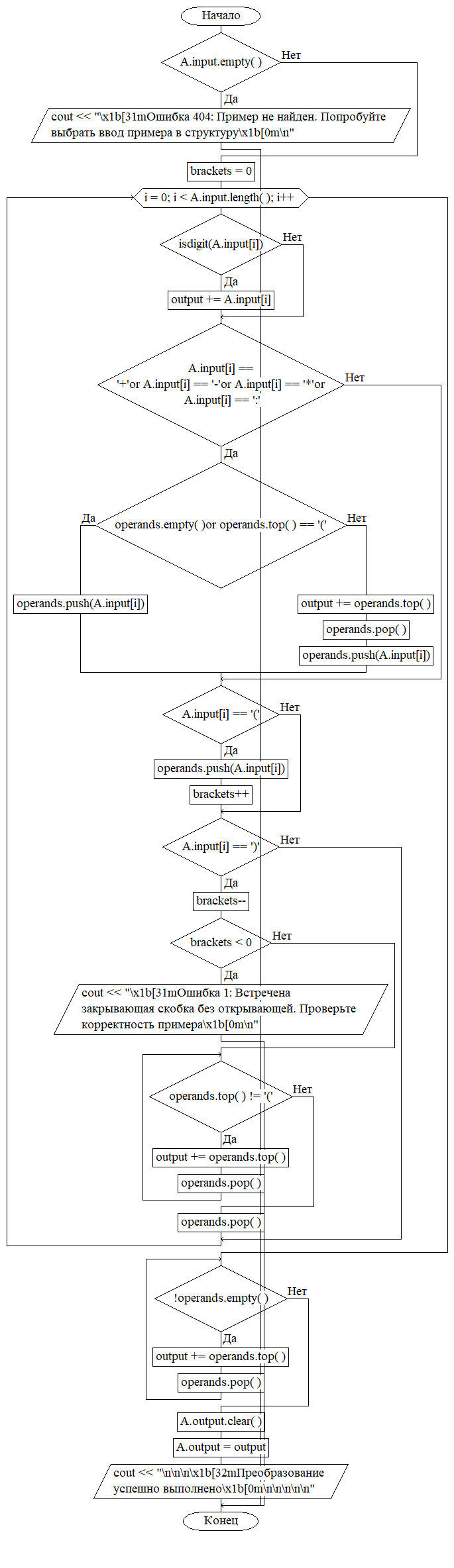
**Void InputS(string s, Post& A)**



**Void Display(Post& A)**



**Void OutputS(Post& A)**

****